EFFICIENT ALGORITHMS FOR TIMING ANALYSIS AND SIGNAL INTEGRITY FOR VLSI CIRCUITS

Ph.D. SYNOPSIS

Submitted by

SUSHOBHIT SINGH

Under the Supervision of

Prof. A. K. Saxena
Supervisor

Prof. P. K. Kalra
Co-supervisor

Prof. D. S. Mishra
Dean, Faculty of Engineering

Prof. V. Prem Pyara
Head Deptt. of Electrical Engineering

Dayalbagh Educational Institute
Dayalbagh, Agra- 282005
2012

Introduction

The exponential growth in complexity and size of VLSI circuits with the technological advancements has posed several problems in its design, modeling, analysis and implementation. The increased functional requirement of a module has caused already straining deep sub-micron node to shrink into nanometer scale, leading to the requirement of more complex and quantitatively large designs. To analyze, implement and optimize these large circuits, computer simulations and static analysis are performed by the VLSI CAD tools. These tools model the circuit connectivity using discrete data structures like graphs and hyper-graphs and then perform the required simulations on the generated graph structure. Due to the complexity and size of the circuits, typical capacity issues are faced by the CAD tools, as it is very difficult even for the most efficient CPUs and memory models to analyze and optimize the designs on single processor. To enhance the scalability of the process, a circuit is divided into several sub circuits, by applying suitable graph partitioning techniques on the original circuit graph and then analyzing and optimizing these circuits in parallel on different processors to achieve the desired results. The objective of selecting a suitable partitioning technique for circuit analysis is to optimize the inter-processor communication process.

Timing Analysis is an analysis technique used to verify the timing behavior of a VLSI circuit. Timing analysis requires computing and propagating the component and inter-connects delays through the circuit. Timing tools models the circuit by a specialized graph structure popularly called as timing graph wherein, the entire circuit is categorized into timing paths and each of the timing paths are tried to be optimized and kept within stipulated timing

constraints, applied by the design environment. Timing analysis can be performed both in static and the dynamic mode, but dynamic analysis is a cumbersome process and requires to generate a response to a lot of test vectors for each timing path in the design leading to slower turn-around time. Whereas, static analysis is performed using worst delay case and gives a faster response while maintaining the desired quality comparable to that of dynamic analysis. A detailed discussion on static timing analysis is presented in [1] [2] [3].

Due to ever shrinking geometries, inter-connects on a single metal layer are placed close to each other leading to significant coupling capacitance between inter-connects. Due to high frequency of signals, the capacitive coupling manifests more rigorously and leads to a high cross-talk between the signals on the coupled nets. This causes the weaker signals to distort further and sometimes leading to timing failure, thus affecting the signal integrity. If signal integrity is not maintained and timing analysis does not take account of signal perturbations due to cross-talk, the final circuit timing behavior may not meet the requirements. Rahman, Yan and Abubakar [13], have pointed out the importance of signal integrity in the digital designs followed by the factors affecting the signal integrity in present state of the art digital designs, further they outlined techniques and tools for analyzing signal integrity. O. Cudert in [31] has pin pointed the problems that are faced by physical design closure in the deep submicron arena, due to signal cross-talk and capacity issues posed by large design sizes.

In this thesis, efforts will be made to develop efficient algorithms to partition the given network into suitable partitions, to ensure minimal inter-processor communication, maintaining the timing schedule and signal integrity.

Literature Review

Graphs and hyper-graphs are discrete data structures that are used to model, a host of problems in the fields of science and engineering. Hyper-graph, by definition is more flexible and can be used to model the real life problems more closely than graphs and can be used for problems like VLSI circuit design, traffic scheduling, compilers, etc. In set theoretic sense, graphs and hyper-graphs are collections of subsets of a given set of objects. The objects in this set are called vertices, while the subsets are called the edges or hyper-edges, respectively for graphs and hyper-graphs. Cardinality of any edge in a graph is two, while a hyper-edge can have any cardinality greater than zero, up to the cardinality of vertex set. Detailed discussion on the graph and hyper-graph theory is presented in [23].

Partitioning is the process of decomposition, which is widely prevalent in almost all the disciplines of science and engineering. Once a computation requirement is modeled as a graph, partitioning can be performed to decompose the problem into graph partitions, these partitions have reduced size and can be used to perform the required computations efficiently in parallel. The objective of graph partitioning, loosely, is to evenly distribute the computation on a set of processors. This is performed by creating several equally weighted sets of graph vertices, while minimizing inter-process communications represented by edges crossing between partitions. Standard

graph partitioning approach, that requires to model the problem into a graph structure and partitioning the vertices of the graph into equally weighted partitions through edge cutting, has been used historically and is employed in software packages such as Chaco [24] and METIS [25]. B. Hendrikson and T. G. Kolda [21], have criticized the standard graph partitioning approach, on the shortcomings of the model itself, which lacks the expression, and also the edge cut metric, that does not take into consideration the communication volume, message passing latencies and difference in processor performance. They further surveyed graph partitioning models alternative and complimenting the shortcomings of the standard model. They described bipartite model that can be used to describe two computational operations, but no more and the solution is to go for k-partite. Further, more general models like hyper-graph model and multi constraint graph models are also described in literature [21], and their merits are demonstrated for several problem solutions. Karypis and Kumar [25], have discussed multilevel graph partitioning algorithms, and proposed a new fast multilevel graph partitioning algorithm for irregular graphs, on the basis of size per computational node.

In VLSI design flow, process of logic synthesis generates a circuit net-list, comprising of standard cells and inter-connects from a high level design description [28]. The circuit net-list is represented as a hyper-graph with components as vertices and inter-connects as hyper-edges. All the analysis and optimization tools analyze and optimize the design using this structure to achieve desired design parameters. To make it possible to complete the design in time and reduce the time to market, the circuit net-list is partitioned by physical design tools as geometric partitions, each of these partitions are

analyzed and optimized in parallel by different design groups, and then assembled back as a complete design to perform a full circuit analysis. A detailed review of net-list partitioning problem and its solutions is provided in [18] [19] [20]. Somasundaram in [10] has proposed a retiming based multi-level graph partitioning algorithm to reduce the worst case circuit delays, keeping the partition areas minimal. In retiming, registers are moved back and forth to get an optimal worst case circuit delay without changing the combinatorial logic.

For performing static timing analysis on a VLSI circuit, the worst case delays of each component on a timing path and inter-connects is required to be calculated and then propagated along the path on the timing graph. The delay for a component is based on the signal transition slope from the last stage and the output load of the component itself, thus to know the exact delay of a component the real transition from the last stage is needed. With the advent of circuit partitioning for large circuit systems, it was deemed important to develop methods for deriving the initial conditions for partitioned timing paths in order to reproduce the full circuit timing characteristics for individual partitioned circuits. To cater to this need hierarchical timing analysis techniques discussed in [5] [6] [7] [8] are developed, which works on a reduced circuit. Several methods are used to get the initial conditions for comparable full circuit and partitioned circuit timing. The circuit partitioning is performed by using combined synthesis and layout optimization techniques [31], and size of each partition is kept such that each sub-circuit can be analyzed and optimized on a single processor. Once a good sub-circuit timing is achieved, the sub-circuits are stitched back to perform cross-hierarchy and

non-partitioned path optimization. S. Prasad, et al. [5] have proposed a post assembly timing closure (PAC) methodology, by keeping just the cross hierarchy timing paths and non-partitioned paths for achieving a good timing for the entire circuit. They have further presented a comparative analysis of this technique with ILM (Interface Logic Models) based timing closure flow which performs delay budgeting [9] to get the initial conditions for interface timing paths of the partitions. Hierarchical timing techniques are presently used in almost all the industry standard timing closure and design implementation flows due to promising memory footprints and runtime scalability, but a reliable hierarchical timing signoff is still a problem, due to approximations like delay budgeting as pointed out by authors in [5] and signal integrity issues due to reduced geometries discussed in [13]. Rajagopal et al. [14] have proposed a hierarchical signoff methodology taking into account the signal integrity by budgeting the quantities which are affected by the cross coupling between the signals on different sub-circuits or between a sub-circuit. The proposed budgeted quantities are the data and clock signal transitions, clock latencies, data signal arrival time and clock jitter due to capacitive coupling. Another proposal in [14] is a pessimistic approach to keep the worst case budgets, but this would make closure very tough to achieve. To keep the inter-hierarchy cross-coupling at bay a proposal is made to shield the inter-hierarchy inter-connects in addition to keeping them apart. Mohammed et al. [15] have proposed a hierarchical noise and signal integrity verification and timing signoff flow by extending the ideas in [14], using parasitic resistance-capacitance (RC) information of the hierarchies and employing the same to perform the delay calculation. In a white paper [22] from Cadence Design Systems® timing signoff problem at the nanometer

scale is been discussed. They have described the increasing complexity of signoff quality timing analysis due to aggressive signal integrity issues resulting from low power techniques like multi-Vt cells.

With the technological advancements in parallel computing paradigm and development of state of the art cluster computing systems, parallel and distributed timing analysis are also approached to divide the computation effort between several processors. A. Holder et al. [16] have given a prototype of a timing analyzer, for performing parallel timing analysis on an IBM Blue Gene® Super Computer. They have done a review on the efforts done by various researchers to achieve distributed and parallel timing analysis. They used a parallel architecture of 64 processors on a benchmark circuit, to partition the circuit and then applied parallel STA algorithms with message passing system to perform distributed STA across the processors. In this approach entire design is loaded on each of the processors and then the partitioning is done, this has considerably reduced the partitioning memory footprint and run time performance, but for large industrial designs this will not be very efficient. They have concluded that partitioning of designs for timing analysis would be a bottle neck for future. Thus performing distributed timing analysis would need good and fast partitioning algorithms for minimizing the inter-processor communication. In another effort for parallelizing the timing analysis algorithms done by Y. Deng and S. Mau [17], GPU architecture on CUDA implementation are used. By topologically sorting the timing points on the unrelated timing paths on the timing graph are determined and then arrival and required timing calculations are performed on the underlying GPU

architecture. In this approach an effort to find the unrelated timing paths is used which can make the basis of partitioning the circuit.

In the modern competitive environment and with motivation to provide a compact and efficient circuit design has increased the complexity of the design parameters enormously. This in turn leads to highly complex timing analysis due to change in system design parameters. Therefore existing hierarchical timing analysis is not able to preserve the accuracy of full circuit timing. Hence rigorous efforts are being made in the direction of partitioning the circuits for distributed timing analysis, while considering large inter-processor communication volumes due to suboptimal partitioning decisions.

Motivation and Purpose of This Work

Circuit partitioning is performed in a typical VLSI physical design flow to reduce the circuit size and perform design operations within a desirable time limit. Although very good circuit partitioning algorithms are available in the literature, which partition circuits based on different criterions like area balancing, path delay minimization, functional modules, etc. but they seems to be inefficient for timing analysis, as a component delay is dependent on the last stage transitions and hence initial conditions for partitioned paths are required to be derived. Though hierarchical timing analysis techniques resolved this problem to an extent and gave promising memory footprints and runtime scalability, but inherent approximations and model inaccuracies make it very difficult to realize desired signoff quality timing analysis. Besides timing approximations, problems like signal cross talk due to smaller geometries are further roadblocks for signoff quality hierarchical timing analysis. This leads to

a painful conclusion that it is not possible to have a highly reliable signoff quality timing analysis done by reducing the circuit using partitioning, unless efficient graph partitioning algorithms are developed to reduce the inter partition dependency for timing analysis. There is thus a need to study various factors effecting timing and generate high degree timing independent sub-circuits to perform scalable and accurate timing analysis.

In this work we would target developing graph partitioning algorithms to reduce the inter-processor communication for performing timing analysis in presence of signal integrity verification. It is therefore proposed to develop algorithms for reliable and efficient graph and hyper-graph partitioning, to obtain

1. Optimal parallel and distributed timing analysis.
2. Reliable signal integrity verification.

The algorithms developed will be tested on some benchmark problems using standard available cluster computing systems.

References

[1] Norman P. Jouppi, “Timing Analysis and Performance Improvement of MOS VLSI Designs”, IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN. , VOL. CAD-6 NO. 4 , pp 650-665 , JULY 1987.

[2] Micahel R. Dagenais, “Timing analysis for MOSFETS : an integrated approach.” , June 1987.

[3] Young Hwan K. et al. , “Electrical-logic simulation and it’s applications.” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. , Vol-8 No. 1 , pp 8 – 22 , Jan 1989.

[4] Y. Blaquiere et al. , “FAST TIMING ANALYSIS OF VLSI CIRCUITS : A DYNAMIC AND HIERARCHICAL APPROACH.” , IEEE International Symposium on Circuits and Systems, Vol-4 , pp 2398 – 2402 , Jun 1991

[5] S. Prasad et al. , “Post assembly timing closure for multi million gate chips.” , 23rd International Conference on VLSI Design, pp 75 – 80 , Jan 2010.

[6] R.H.J.M. Otten, “Timing Closure: the solution and its problems.” , Design Automation Conference Japan, pp 359 – 364 , June 2000.

[7] Jason Cong, "Timing closure based on physical hierarchy", International Symposium of Physical Designs, ISPD, 2002

[8] Wei-Jin Dai, "Hierarchical physical design methodology for multimillion gate chips", International Symposium of Physical Designs, ISPD, timing-closure-driven design method, International Conference on Computer Aided Design, ICCAD, 2000.

[9] Chien-Chu Kuo; Wu, A.C.-H, "Delay budgeting for a timing-closure driven design method", International Conference on Computer Aided Design, pp. 202-207, ICCAD , 2000.

[10] K. Somasundaram, “Multi-Level Sequential Circuit Partitioning for Delay Minimization of VLSI circuits”, Journal of Information and Computer Science. , Vol-2 No-1 , pp 66 – 70 , 2006.

[11] J. Minami , et al. , “A Circuit Partitioning Algorithm Under Path Delay Constraint.” , The 1998 IEEE Asia-Pacific Conference on Circuits and Systems , pp 113 – 116 , Nov 1998.

[12] J. Kim, et al. , “Parallelizing Post-Placement Timing Optimization.” , 20th International Parallel and Distributed Processing Symposium. , pp 10 , June 2006.

[13] T. Rahman, et al. , “Signal Integrity for High Speed Digital Design.” , 9th International Symposium on Antennas Propagation and EM Theory (ISAPE), pp 1113 – 1115 , Nov 2010.

[14] K.A. Rajagopal, et al. , “A Comprehensive solution for true hierarchical timing and crosstalk delay signoff.” , 19th International Conference on VLSI Design. , Jan 2006

[15] S. A. Mohamed, et al. , “A Noise and Signal Integrity Verification Flow for Hierarchical Designs.” , IEEE International Conference on Computer Applications and Industrial Electronics(ICCAIE), pp 250 – 255 , Dec 2011.

[16] A. Holder, et al. , “Prototype for a Large-Scale Static Timing Analyzer running on an IBM Blue Gene”, IEEE International Symposium on Parallel & Distributed Processing, Workshops and PhD Forum (IPDPSW), pp 1 – 8 , April 2010.

[17] Y. Deng and S. Mau, “The potential of GPUs for VLSI physical design automation.” , 9th International Conference on Solid-State and Integrated-Circuit Technology(ICSICT), pp 2272 – 2275 , Oct 2008.

[18] C.J. Alpert, J-H. Huang, and A.B. Kahng , “Recent directions in netlist partitioning”, Integration, the VLSI Journal, 19(1–2):1–81, 1995.

[19] A. Kahng, “Futures for partitioning in physical design.”, Proc. ACM/IEEE International Symposium on Physical Design, pp 190–193, 1998.

[20] J. Cong, M. Romesis, and M. Xie, “Optimality, scalability and stability study of partitioning and placement algorithms”, Proceedings of International Symposium on Physical Design, pp 88–94, 2003.

[21] B. Hendrickson and T. G. Golda, “Graph partitioning models for parallel computing”, Journal of Parallel Computing, Vol-26 Issue 12, pp 1519 – 1534 , 2000.

[22] “Complete Timing Signoff In the Nanometer Era.” (www.whitepapersdb.com and tech-white-papers.com)

[23] Vitaly I. Voloshin. "Introduction to Graph and Hypergraph Theory". Nova Science Publishers, Inc., 2009

[24] B. Hendrickson, R. Leland, The Chaco user's guide, version 2.0, Technical Report SAND95-2344, Sandia National Labs, Albuquerque, NM, 1995.

[25] G. Karypis, V. Kumar, “A fast and high quality multilevel scheme for partitioning irregular graphs”, Technical Report 95-035, Department of Computer Science, University of Minnesota, Minneapolis, MN, 1995.

[26] Xin Sui, et al. “Parallel Graph Partitioning on Multicore Architectures”, Languages and Compilers for Parallel Computing (LCPC), 2010.

[27] G. Karypis, R. Aggarwal, V. Kumar and S. Shekhar, “Multilevel Hypergraph Partitioning : Applications in VLSI Dopmain”, IEEE transactions on VLSI systems, Vol. 7, No. 1, pp. 69-79, March 1999.

[28] H. Kaeslin, “Digital Integrated Circuit Design: From VLSI Architectures to CMOS Fabrication”, Cambrige University Press, 2008

[29] R. B. Hitchcock Sr. et al., “Timing analysis of computer hardware,” IBM Journal of Research and Development, vol. 26, no. 1, pp. 100–105, 1982.

[30] A. B. Kahng, J. Lienig, I. L. Markov and J. Hu, “VLSI Physical Design : From Graph Partitioning to Timing Closure”, Springer, 2011

[31] O. Cudert, “Timing and Design Closure in Physical Design Flow”, IEEE Proceedings of the International Symposium on Quality Electronic Design, 2002.